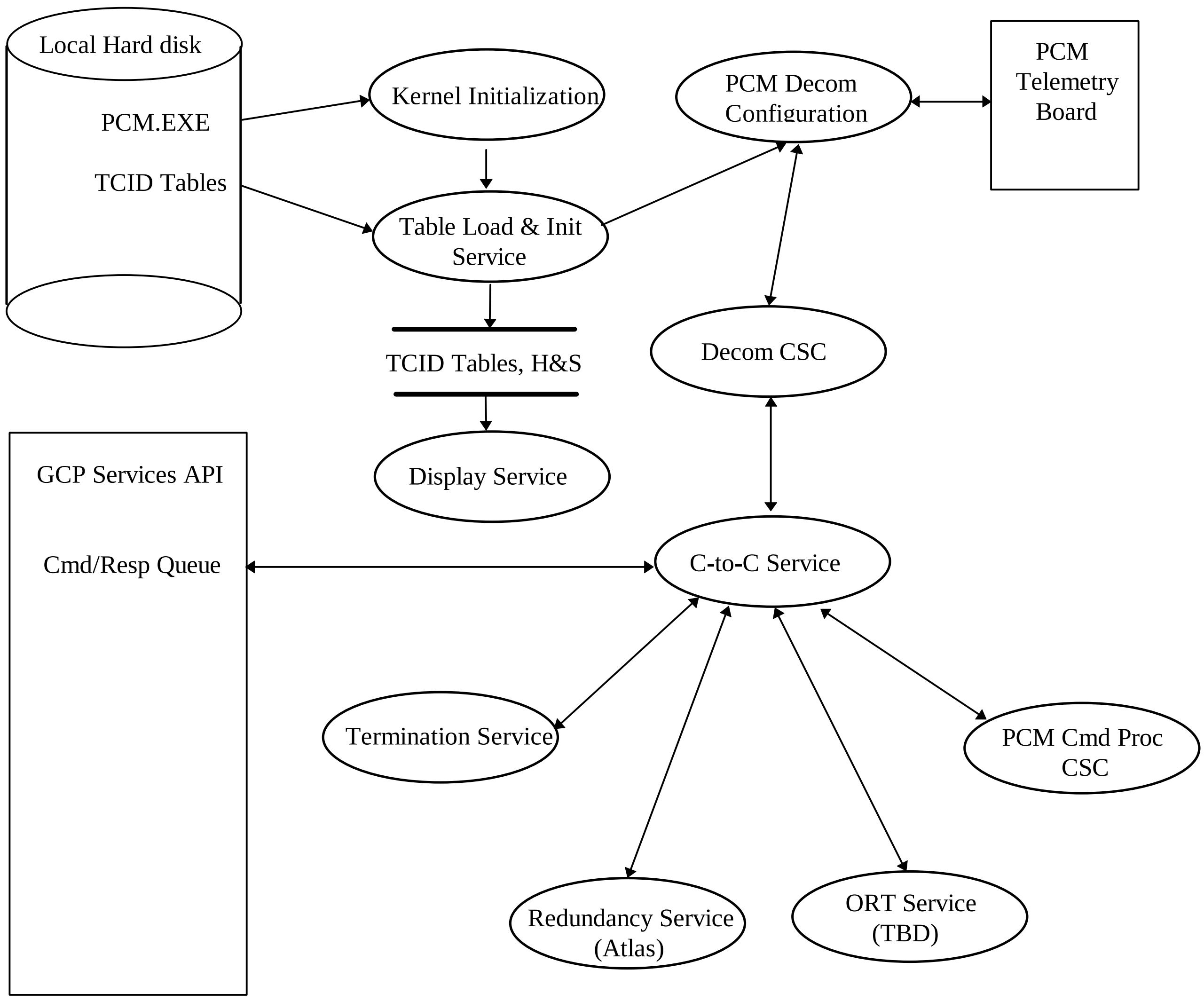
1. PCM PROCESS CONTROL CSC

1.1 PCM PROCESS CONTROL CSC Introduction

1.1.1 PCM PROCESS CONTROL CSC Overview

The Process Control CSC is responsible for initialization/termination of the Front End Process Controller (FEPC) card. This CSC is also responsible for distributing of C-to-C commands to other CSCs.



1.1.2 PCM Process Control CSC Operational Description

The FEPC Initialization will transition through four stages: SCID Initialization, SCID/TCID Load, Ready, and Operational. The FEPC enters the SCID Initialization stage via power-up, manual reset, or by an Init SCID command. If SCID Initialization is successful, the FEPC enters the SCID/TCID Load stage and is ready to accept TCID tables load command or SCID Init command (reboot to go back to SCID Initialization stage). If TCID Load is successful, the FEPC enters the Ready stage. At this point, the FEPC will accept a limited set of commands; one of which is Activate Gateway. An Activate Gateway command will cause the FEPC Initialization to activate other CSCs and transition to Operational mode. Other C-to-C's will then be accepted and routed to appropriate CSC for processing.

1.2 PCM Process Control CSC Specifications

1.2.1 PCM Process Control CSC Groundrules

- The GCP Service API is implemented under the Gateway Common Service CSCI.
- Subsystem Integrity is implemented under the Gateway Common Service CSCI.

1.2.2 PCM Process Control CSC Functional Requirements

1 FEPC Initialization

1.1 The PCM FEPC Initialization shall accept and process the following commands: SCID/TCID Initialization, Activate Gateway, Terminate Gateway.

1.2 The PCM FEPC Initialization shall transition to SCID Initialization mode via power-up, manual reset or by an Init SCID command.

1.3 When in the SCID/TCID Load mode, only SCID/TCID Initialization command shall be accepted.

1.4 The PCM Gateway shall allow multiple revisions of TCID to be stored locally.

1.5 When in the SCID/TCID Load mode or Ready mode, the PCM FEPC Initialization shall process SCID/TCID Initialization to be reassigned to a SCID/TCID configuration.

1.6 The PCM FEPC Initialization shall report an error upon receipt of an PCM Activate Data Acquisition (ADA) command if TCID tables are not loaded.

2 FEPC Termination

2.1 The PCM Gateway shall accept and process request to terminate (Terminate Gateway) the subsystem only when data acquisition is inhibited.

2.2 The PCM Gateway shall notify the system when it receives and processes a terminate command.

2.3 The PCM Gateway shall update status information during termination of subsystem processing.

2.4 The PCM Gateway shall record all termination messages on local storage media.

2.5 The PCM Gateway shall generate a system message prior to self termination of subsystem processing.

2.6 The PCM FEPC shall return to Ready mode when termination is successful.

3 Redundancy (post Thor)

3.1 The PCM Gateway shall provide the capability to be configured as part of an active/standby pair.

3.2 The active PCM Gateway shall provide the capability to request an active/standby switchover when a failure renders the active PCM Gateway incapable of supporting operations.

3.3 Only the active PCM Gateway shall write command and/or measurement data to the RTCN.

3.4 Both active and standby PCM Gateways shall maintain health and status in the RTCN.

3.5 The active PCM Gateway shall send tables to the standby PCM Gateway at load time.

3.6 The PCM Gateway shall maintain table sync between the active and standby PCM Gateways.

3.7 Both the active and standby PCM Gateways shall receive Test End Item (TEI) data.

3.8 Only the active PCM Gateway shall issue a response to a command request.

3.9 A demand checkpoint update shall be discontinued when a switch-over from Active to Standby occurs.

3.10 Switch-over from the active to standby PCM Gateway shall be accomplished with no loss of commands.

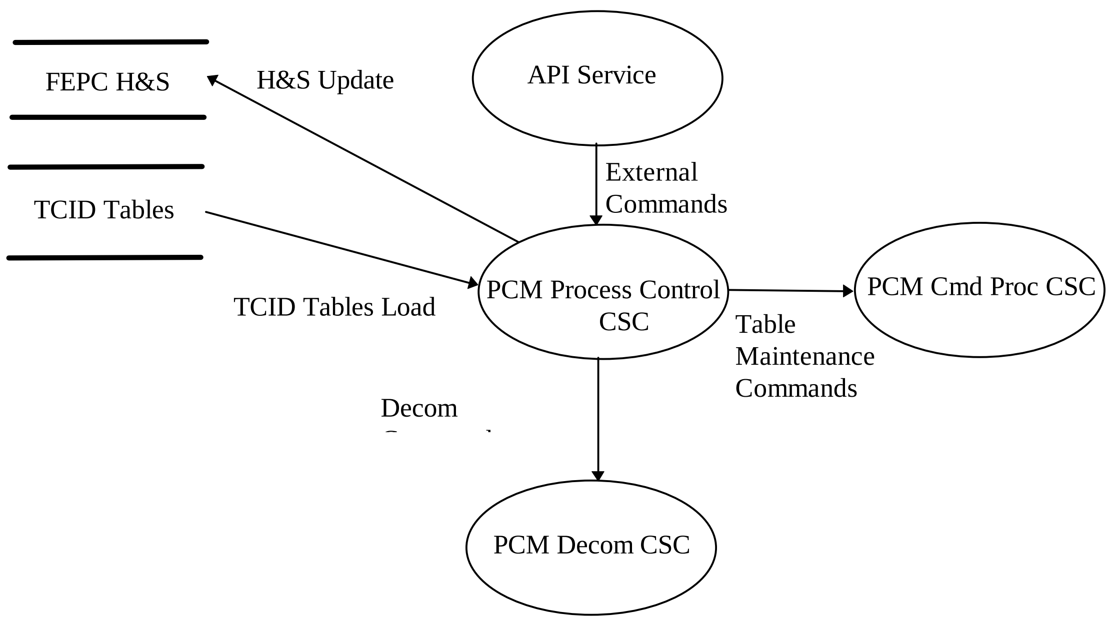
3.11 Any PCM Gateway shall be capable of operating independently of all other PCM Gateways even if configured as part of an active/standby pair.

3.12 The active PCM Gateway shall request an active/standby switchover and continue to process data when the number of consecutive PCM sync errors reaches the sync error threshold and the standby is still in go mode.

1.2.3 PCM Process Control CSC Performance Requirements

- The PCM Gateway shall respond to a C-to-C command within 10 milliseconds per “Command Management to GSE Gateway IDD” (84K00353), section 3.2.2 Timing Constraints.

1.2.4 PCM Process Control CSC Interfaces Data Flow Diagrams



The PCM Process Control CSC will receive external commands (i.e. C-to-C) via an Application Program Interface (API) library call.

When in the SCID/TCID Load stage, if a TCID Init command is received, via an API call, the PCM Process Control CSC will loads TCID tables form the local hard disk.

When in the Operational stage, the PCM Process Control CSC will distribute C-to-C commands to either PCM Command Processing CSC or PCM Decommutation CSC based on their routing codes.

FEPC Health and Status block will also be updated.

Note: This is the end of the Design Panel 2 Required material. The information covered below is for Design Panel 3.

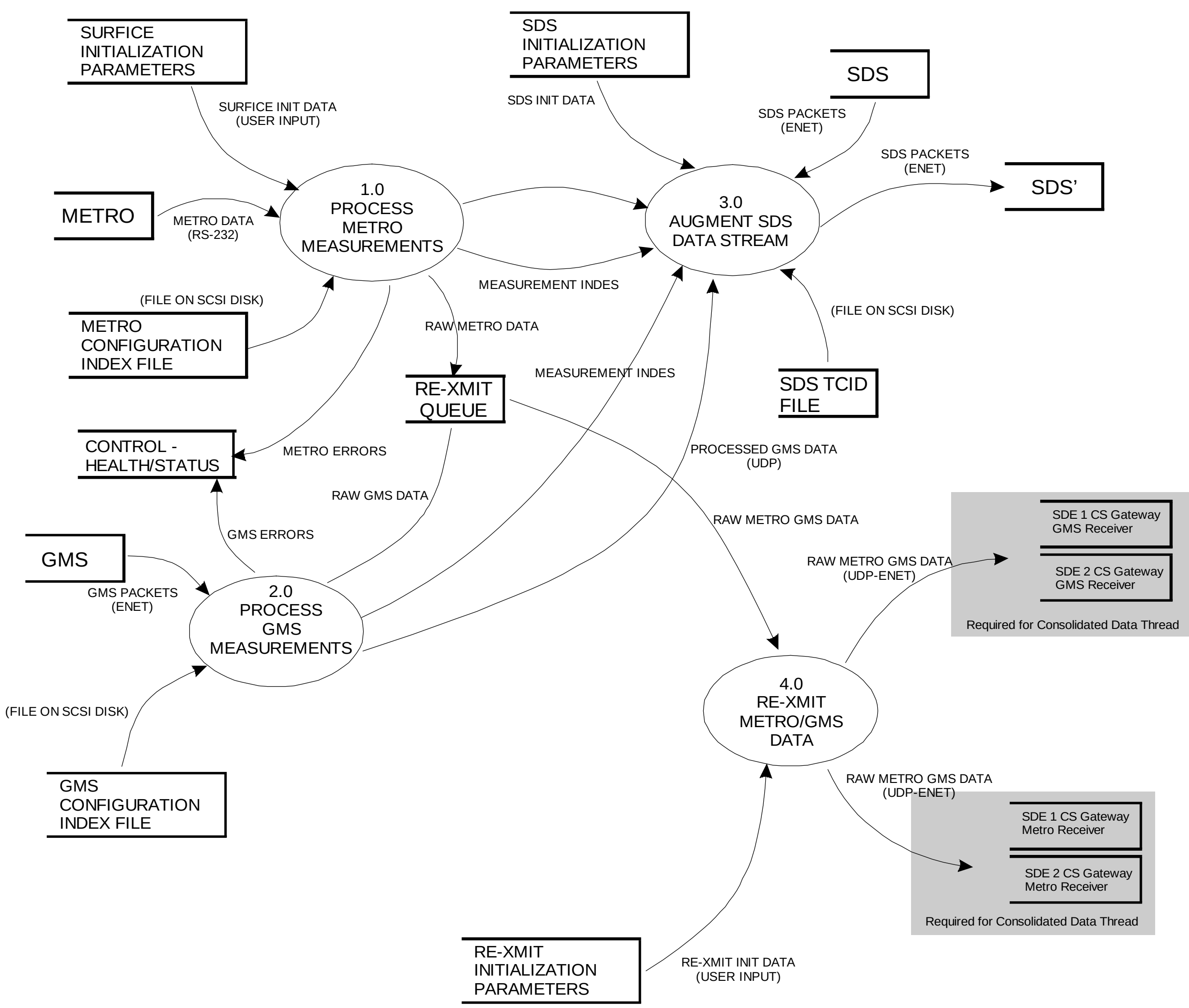
1.3 CSC Name Design Specification

Include a brief description of the architecture of the CSC. If there are any priorities associated with the design they should be specified here.

1.3.1 CSC Name Detailed Data Flow

This data flow provides a pictorial representation of the data flow between external sources and destinations and the major and minor functions of the CSC. This is an example detailed data flow diagram.

Detailed Data Flow Diagram Example



The purpose of the Detailed Interface/Data Flow diagram is to show all of the interfaces, internal as well as external, of the CSC. It conveys in a pictorial format all of the input and output streams that the CSC deals with, but not their content. The content of each of these streams of data is described in the detailed design below. Include with the diagram a short paragraph describing the data flow so that a reader can pick up the document and understand the data flow without a conversation with the developer.

Note: Do not use the underscore_character between words. This is an English language document that we want people to be able to pick up and read in their natural language, not a pseudo programming language.

1.3.2 CSC Name External Interfaces

1.3.2.1 CSC Name Message Formats

This data is the System Messages output by the CSC.

Example:

Message Number = ________
Message Group = _________
Severity = __________

GSE Gateway **I1** Adapter Failure - Polling Stopped - Switchover Requested
Status Register Contains H**#I2#** S/B H**#I3#**

Insert #1 = One Character Integer Value 1 through 8
Insert #2 = Four Character Hex Value
Insert #3 = Four Character Hex Value

Help Information Content:

Help information content will contain a breakdown of the status register showing the meaning of each bit. This file would also show (in time) the likely cause of different failure indications.

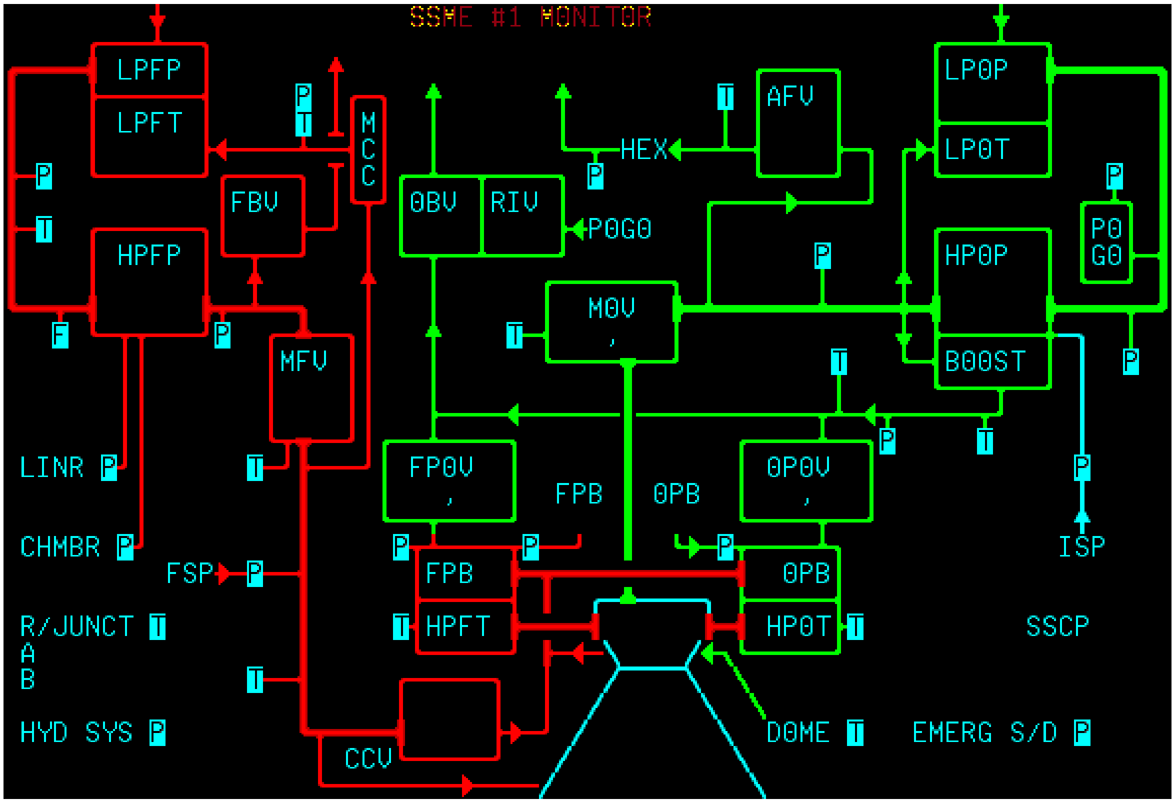
Details Information:

Details Information will be available on the Status Register only.

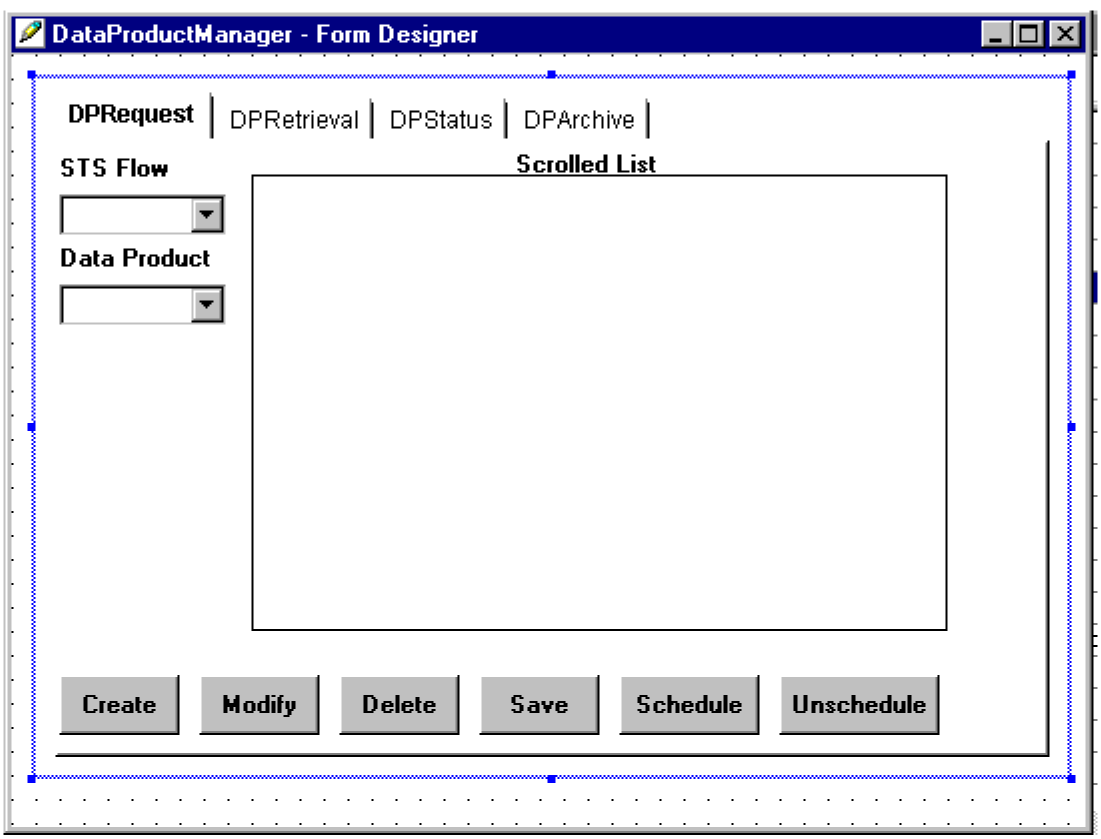
1.3.2.2 CSC Name Display Formats

This is the design of Displays produced by the CSC.

Example 1:


SSME #1 MONITOR
LPFP
LPFT
FBV
MCC
OBV
RIV
POGO
HEX
AFV
LPOP
LPOT
HPFP
MOV
HPOP
PO GO
BOOST
MFV
LINR
CHMBR
FSP
R/JUNCT
A
B
HYD SYS
FPOV
FPB
OPB
OPOV
FPB
OPB
HPFT
HPOT
ISP
SSCP
CCV
DOME
EMERG S/D

Example 2:

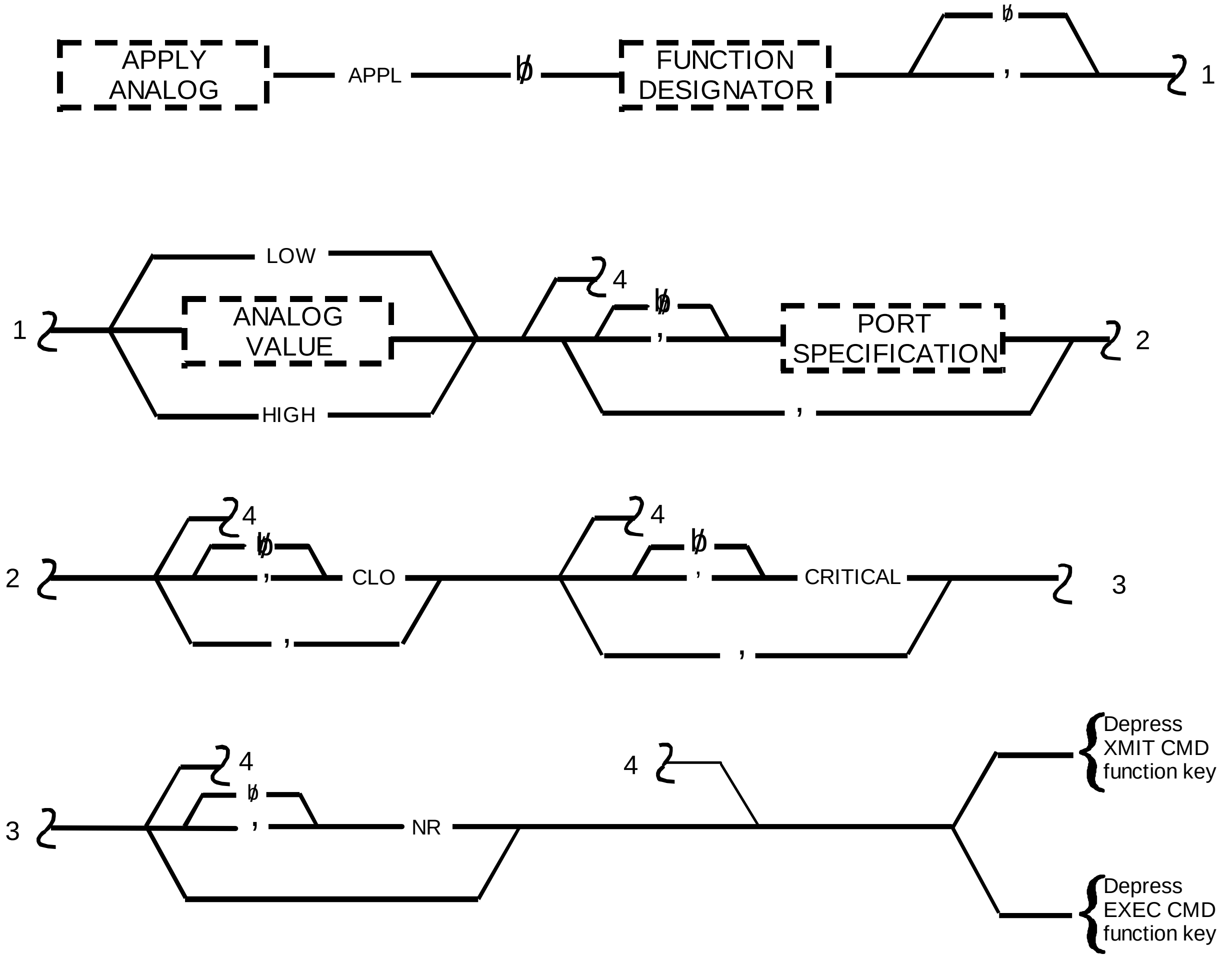

DataProductManager - Form Designer
DPRequest
DPRetrieval
DPStatus
DPArchive
STS Flow
Data Product
Scrolled List
Create
Modify
Delete
Save
Schedule
Unschedule

1.3.2.3 CSC Input Formats

This data is for CSCs that have a language like interface (e.g., Command Processor — Apply Analog)

Example:

APPLY ANALOG



1.3.2.4 Recorded Data

This area contains a table of data that is recorded by the CSC. All data being recorded must be contained in the table. List the name of each message recorded, the type of recording (listed in the example), and location the data is recorded to. Data will be recorded on the local storage device only when approved by the Design Panel. These approvals can be expected to be rare.

Name of Recorded Data	Recording Type	SDC	Local
		X	
		X	
		X	
		X	
		X	

1.3.2.5 CSC Name Printer Formats

This data documents the design of anything printed on printers.

Example:

LDB MASS MEMORY READ
CDT=-99:2359/59 GMT=132:1232/24.375 GATEWAY=LDBA CMD=MASS MEMORY READ FCNL DEST=MM TRNS ID=
FTSB= 5 1 4 4 DEST=MM 1 SEQ BLKS=9

MASS MEMORY RESPONSE TIME=1232/24.531 GMT FUNCTIONAL DESTINATION= MM TRANS ID= 01E7 CA
MM2 FILE 0 TRACK 6 SUBFILE 7 BLOCK 0 BLOCK# 30 0F 31 BLOCKS READ ERR

```
0000   C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6   C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6   000F
0010   C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6   C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6   001F
0020   C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6   C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6   002F
 •
 •
01F0   C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6   C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6   01FF
```

LDB MASS MEMORY WRITE
CDT=-99:2359/59 GMT=132:1245/02.465 GATEWAY=LDBA CMD=MASS MEMORY WRITE FCNL DEST=MM TRNS ID
FTSB= 5 1 4 4 DEST=MM 1 SEQ BLKS=3 DATA:
0000 0000

MASS MEMORY RESPONSE TIME=1245/02.900 GMT FUNCTIONAL DESTINATION = MM TRANS ID=02F0
CAPABILITY 1 WRITE TO MM1 **UNSUCCESSFUL**

LDB MMU READ LOAD BLOCK
CDT=-99:2359/59 GMT=132:1247/24.449 GATEWAY=LDBA CMD=MMU READ LD BLCK FCNL DEST=MM TRNS ID=
PH/LD BLK = #0302
MM VERSB=#0000 PATCH ID=#0000 FSW ID= 8 CHKSM #43A9 MM1 NOCOMP DUMP

MASS MEMORY RESPONSE TIME=1247/12.950 GMT FUNCTIONAL DESTINATION = MM TRANS ID=02A7
CAPABILITY 2 READ & DUMP LOAD BLOCK MM1 PHASE=01 LOAD BLOC0A W/O COMPARE
STATUS= SUCCESSFUL
FILE 3 TRACK 6 SUBFILE 1 BLOCK 9 BLOCK COUNT= 31

```
0000   9E31 B5F6 0001 C6C6 C6C6 C6C6 C6C6 C6C6   C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6 C6C6   000F
 •
```

01F0	A1F3	3089	A1F3	C6C6	C6C6	C6C6	C6C6	C6C6	C6C6	C6C6	C6C6	C6C6	C6C6	C6C6	C6C6	C6C6	01FF

1.3.2.6 Interprocess Communications (C-to-C Communications?)

This is the data that is sent between CSCs that may or may not be in the same processor. Examples are the C-to-C formats that Tom Jamieson has developed for us. These formats are prepared initially by the Systems Engineering Team (Tom Jamieson) but not owned by the SEI group during the design of the CSCs that send and receive them. The SW design team is expected to

1) Assume control of the design of the C-to-C
2) Develop the design
3) Document the design — in this document?
4) Provide the design to Tom Jamieson after DP3 approval for incorporation into the System Interface Document (TJJ is this name correct?)

Example:

Change Data Packet Payload Body Entry (Analog)

B15 B14 B13 B12	B11 B10 B9 B8 B7 B6 B5 B4 B3 B2	B1 B0
0000-1001 *	Status	FDID MSB
FDID - 16 LSB		
CC		
EU**		
EU**		

* = "this is an analog entry and 0000-1001 = 100 usec time".
** = 32 bits

1.3.2.7 CSC External Interface Calls (e.g., API Calling Formats)

This is the data that is sent between CSCs via a calling mechanism (e.g., API call)

Example:

TBS

1.3.2.8 CSC Name Table Formats

This data documents the design of the tables used internal to the CSC and provided from an outside source (e.g., Gateway Table Build, OLDB, etc.)

Example:

TBS

1.3.3 CSC Name Test Plan

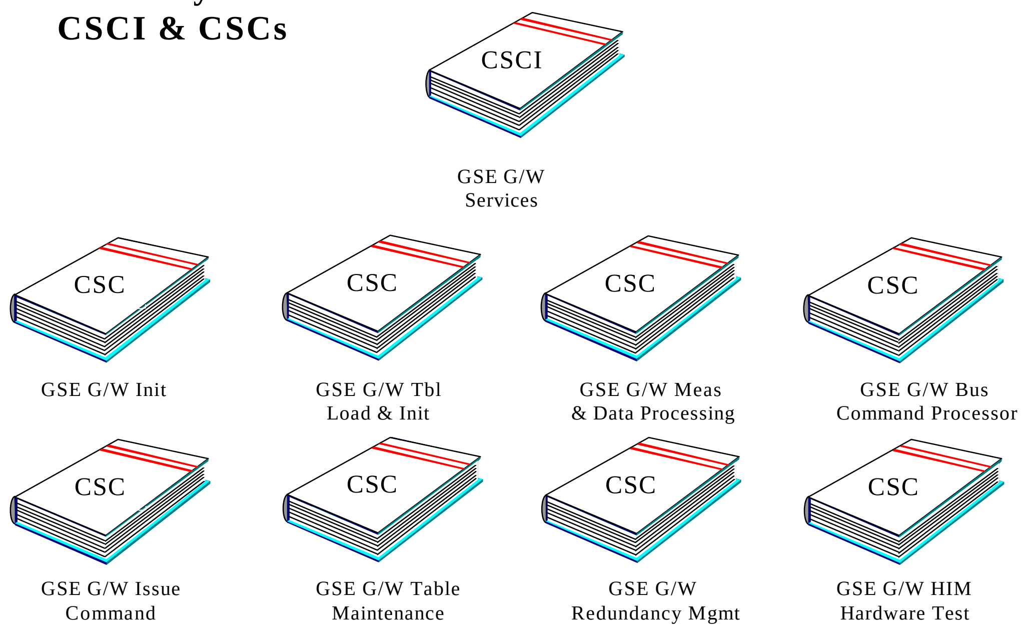
Example:

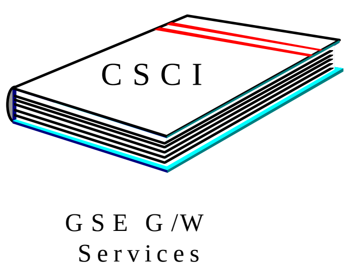
TBS

Appendix A

In most cases a CSCI is composed of the functionality contained in multiple CSCs that are part of the CSCI. The intent of the SRS/Design Specification is to capture the requirements and design of both the CSCI and its CSCs in one set of volumes. Since we are using electronic forms of documentation for the most part these should be easily captured in one place with links to others? At any rate one option for the structure of a CSCI document is as follows:

GSE Gateway Services
CSCI & CSCs



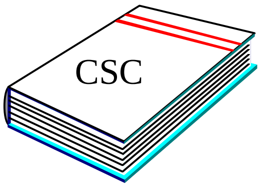


CSCI Outline

1.0 CSCI Name
1.1 CSCI Name Introduction
1.2 CSCI Name Overview
1.2.1 CSC 1 Name Document
1.2.2 CSC 2 Name Document
1.2.3 CSC 3 Name Document
o
o
o
1.2.N CSC N Name Document

Design Panel 2

CSC Requirements Spec

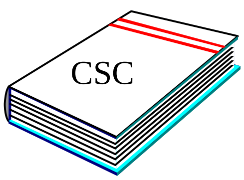


GSE G/W Init

1.0 GSE G/W Init
1.1 GSE G/W Init Introduction
1.1.1 GSE G/W Init Overview
1.1.2 GSE G/W Init Operational Description
1.2 GSE G/W Init Specifications
1.2.1 GSE G/W Init Groundrules
1.2.2 GSE G/W Init Requirements
1.2.3 GSE G/W Init Performance Requirements
1.2.4 GSE G/W Init Interfaces/Data Flow Diagrams

Design Panel 3

CSC Requirements Spec



GSE G/W Init

1.0 GSE G/W Init
1.1 GSE G/W Init Introduction
1.1.1 GSE G/W Init Overview
1.1.2 GSE G/W Init Operational Description
1.2 GSE G/W Init Specifications
1.2.1 GSE G/W Init Groundrules
1.2.2 GSE G/W Init Requirements
1.2.3 GSE G/W Init Performance Requirements
1.2.4 GSE G/W Init Interfaces/Data Flow Diagrams

Present only Changes

CSC Design Spec

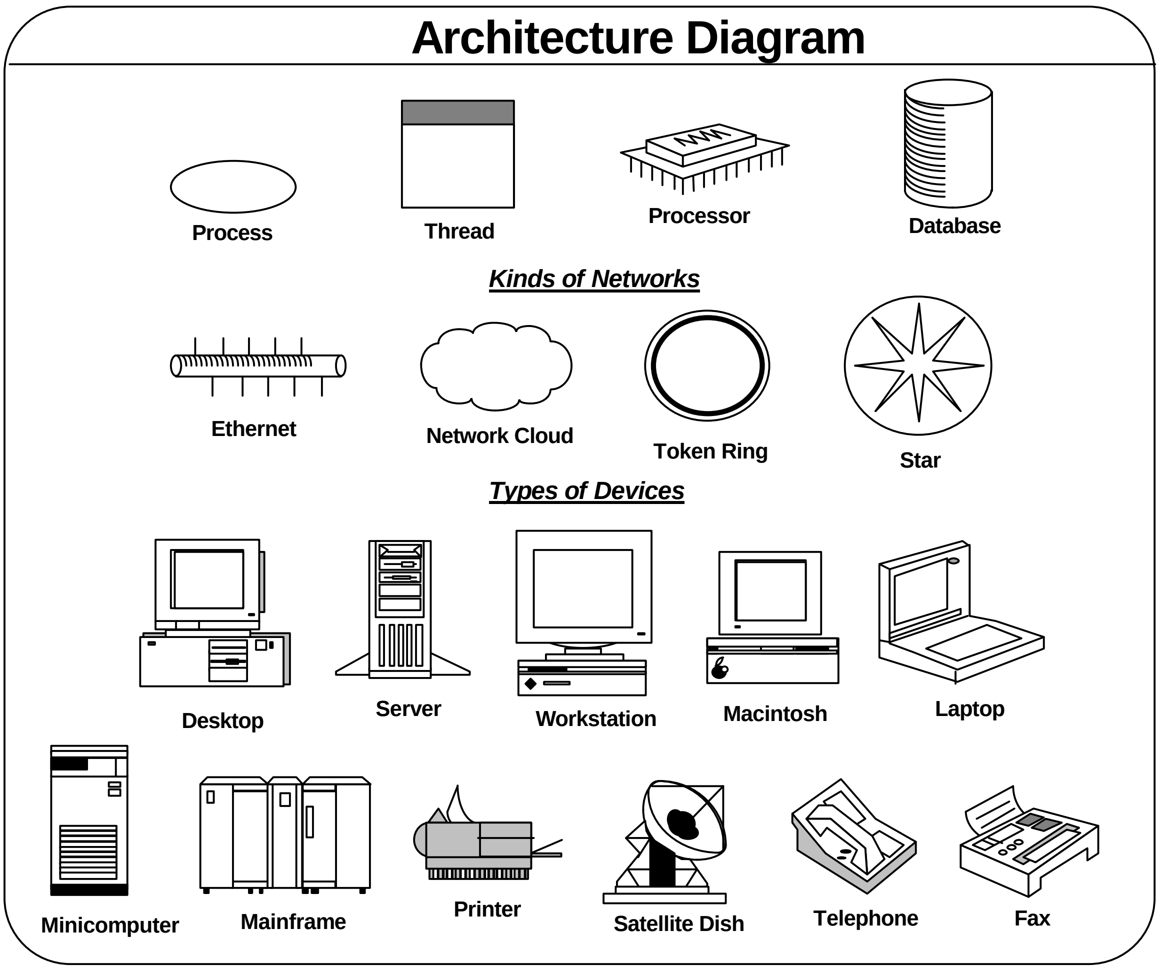
1.3 GSE G/W Init Design Specification
1.3.1 GSE G/W Init Detailed Data Flow
1.3.2 GSE G/W External Interfaces
1.3.3 GSE G/W Internal Interfaces
1.3.4 GSE G/W Structure Diagram
1.3.5 GSE G/W Test Plan

Appendix B

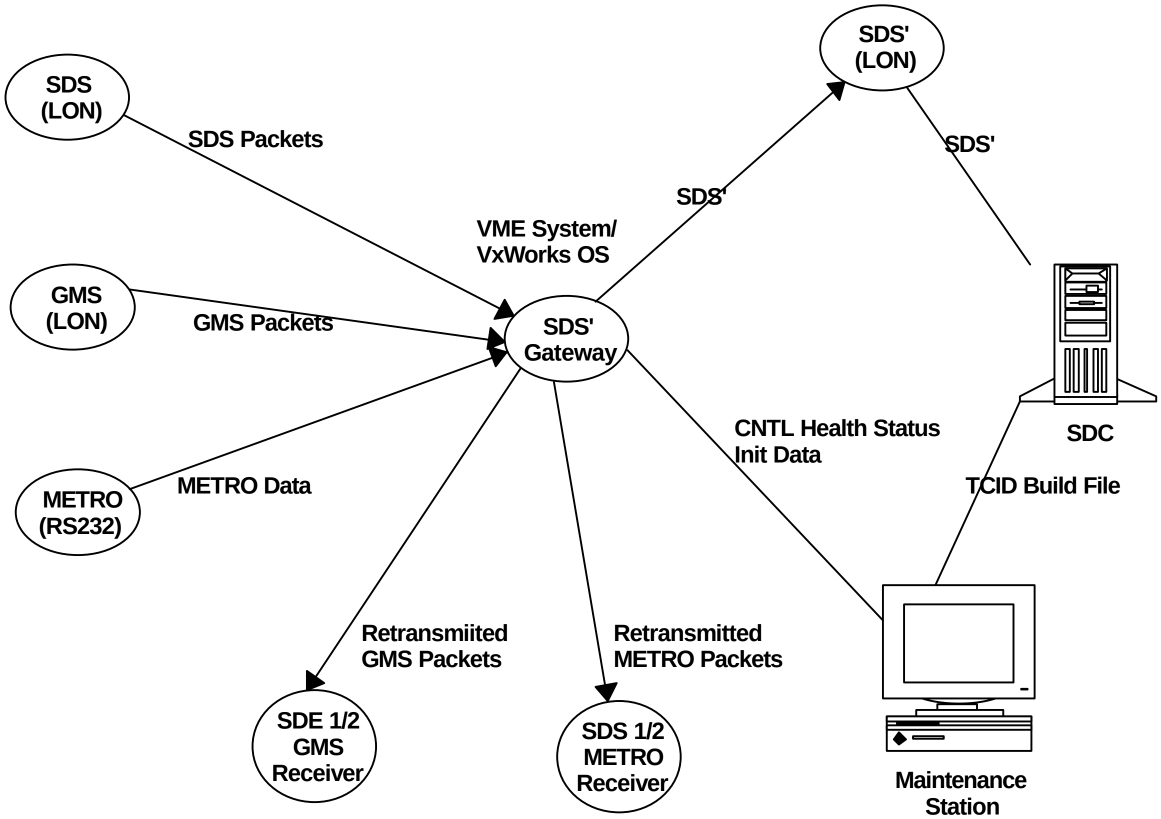
This appendix provides information that will be used for CSCs that are developed using Object Oriented Design (OOD) methodology.

Note: The information in Appendix B is preliminary and will change as the OOD methodology to be used on the CLCS project matures.

The OMT/Rumbaugh notation will be used to document the architecture according to the following legend:



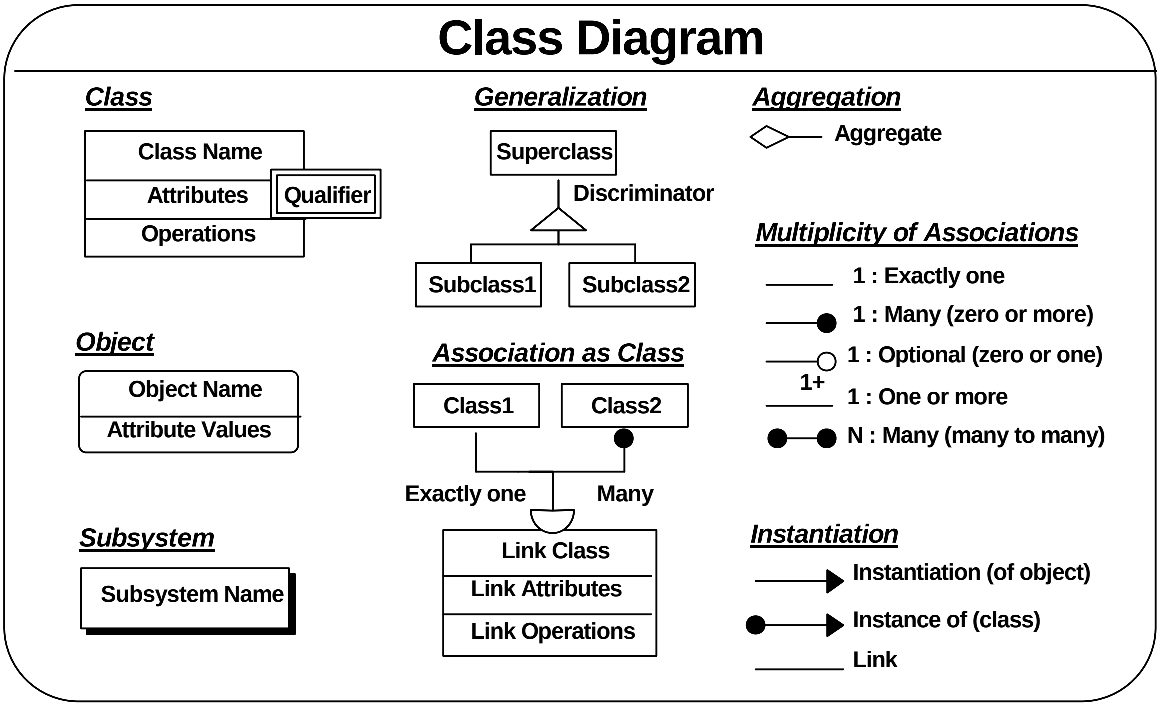
An example architecture diagram is shown below:



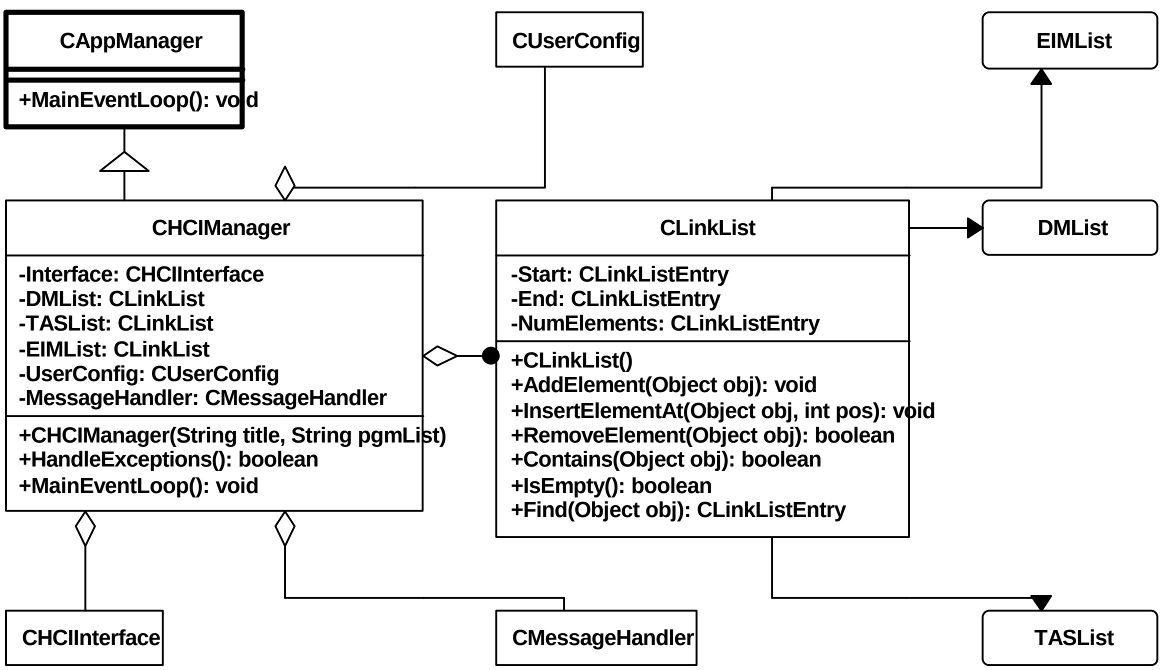
1.4 DP2

Prior to DP2, the designer will decide whether to use Structured Design or OOD. If structured design is to be used, the examples used in the body of this document will be used. If OOD is to be used, the CSC designer should provide a set of preliminary class diagrams. At DP2, the focus is on requirements and preliminary design. Accordingly, the class diagrams presented at DP2 should describe the key abstractions in the requirements ("problem space" classes).

The OMT/Rumbaugh notation will be used to document class information according to the following legend:



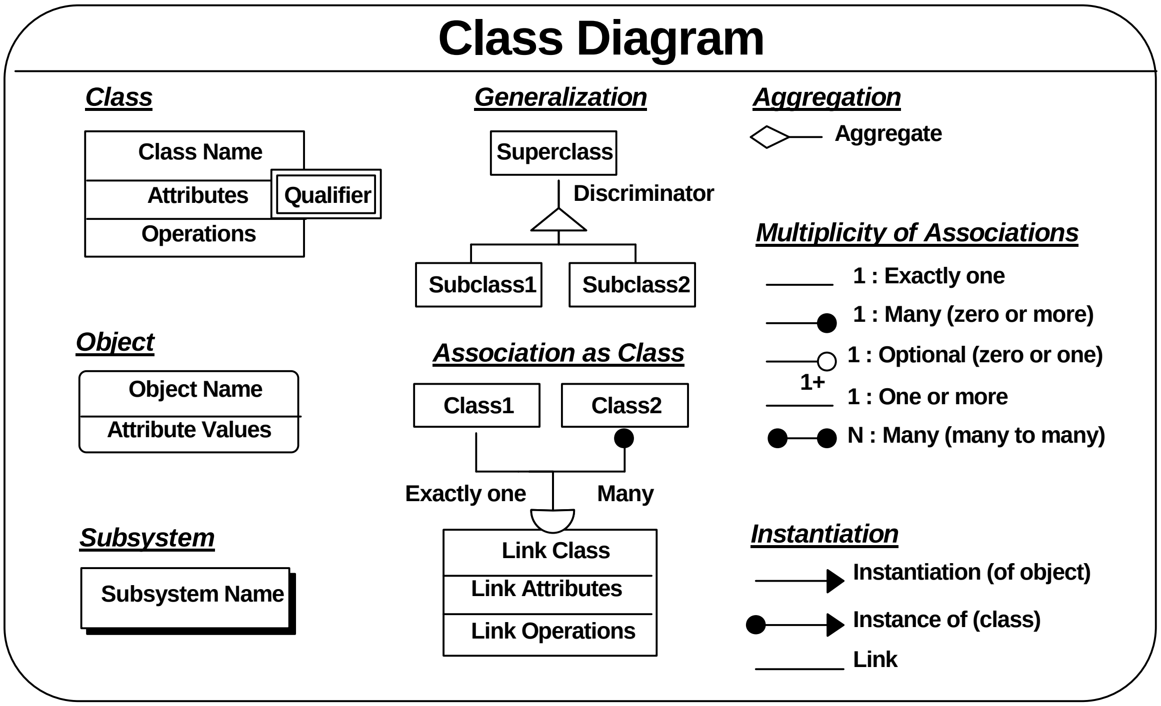
An example class diagram is shown below:



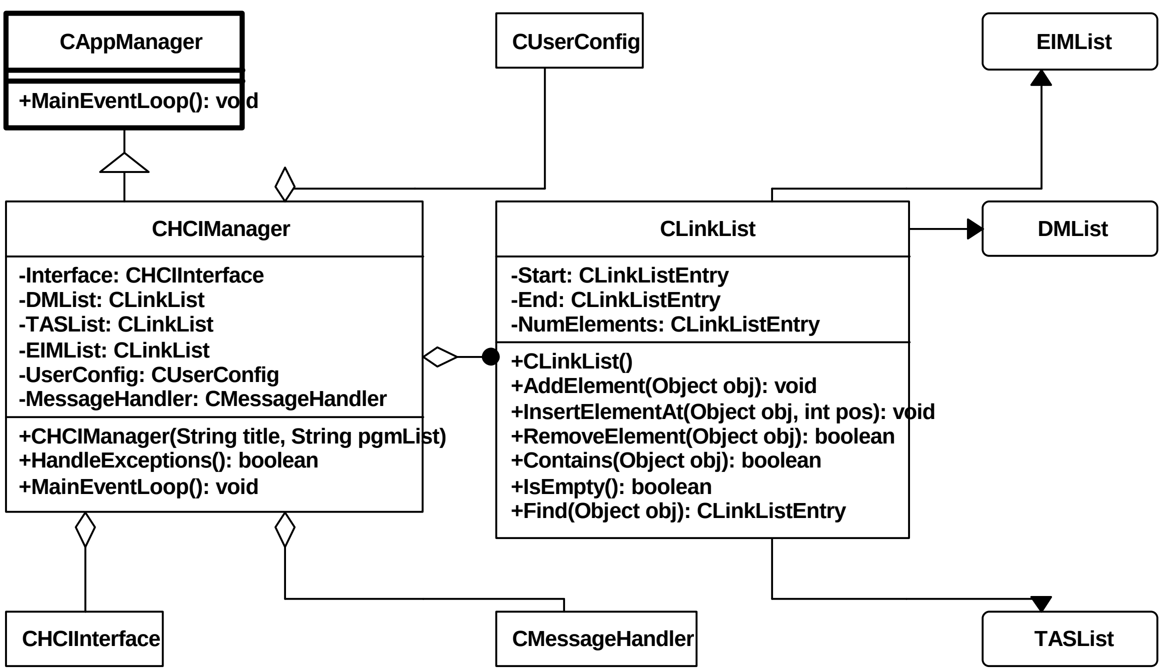
1.5 DP3

For DP3, the designer should provide a set of additional class diagrams that augment the analysis phase classes presented at DP2. Further refinements of the classes presented at DP2 may be included to provide context and aid in understanding. The purpose of these diagrams are to document the mechanisms that will be used to implement the requirements for this CSC.

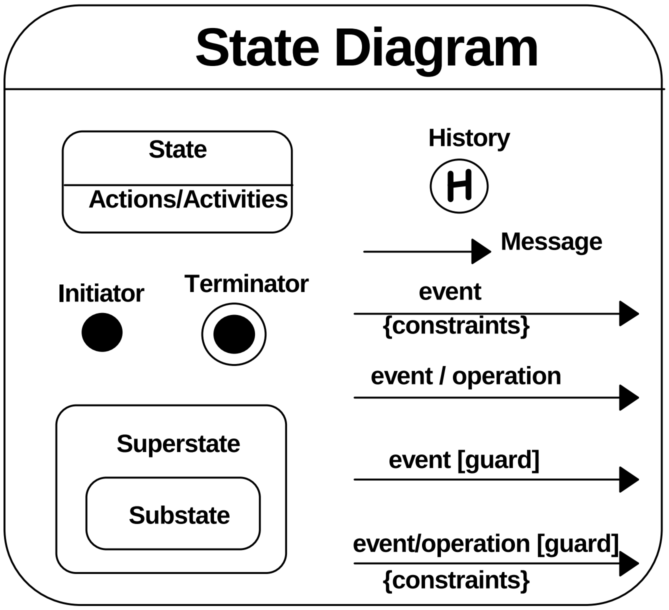
The OMT/Rumbaugh notation will be used to document class information according to the following legend:



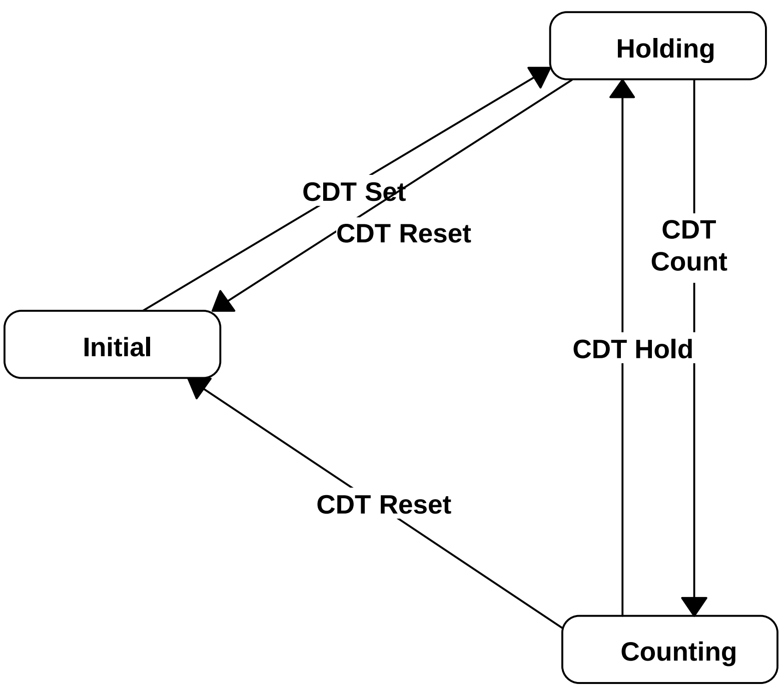
An example class diagram is shown below:



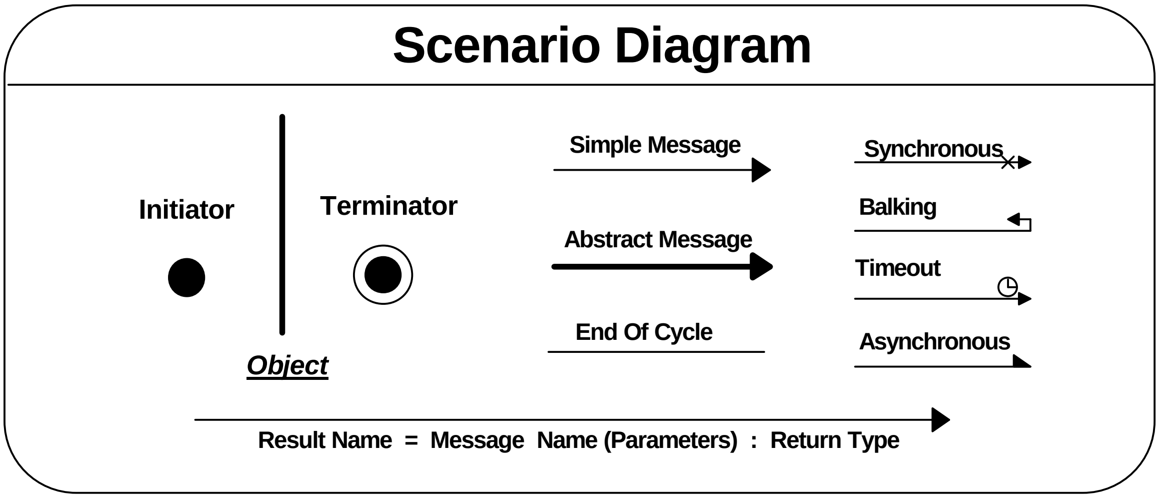
In addition, for classes that exhibit state machine behavior, a state diagram should be included that depicts the dynamic behavior of the class according to the following legend:



An example state diagram for a software implementation of a countdown clock is shown below:



In addition, key scenarios of the CSC's operation or key processing threads should be documented with an scenario diagram according to the legend below:



An example scenario diagram that illustrates the message flow responsible for opening a specific valve is shown below: